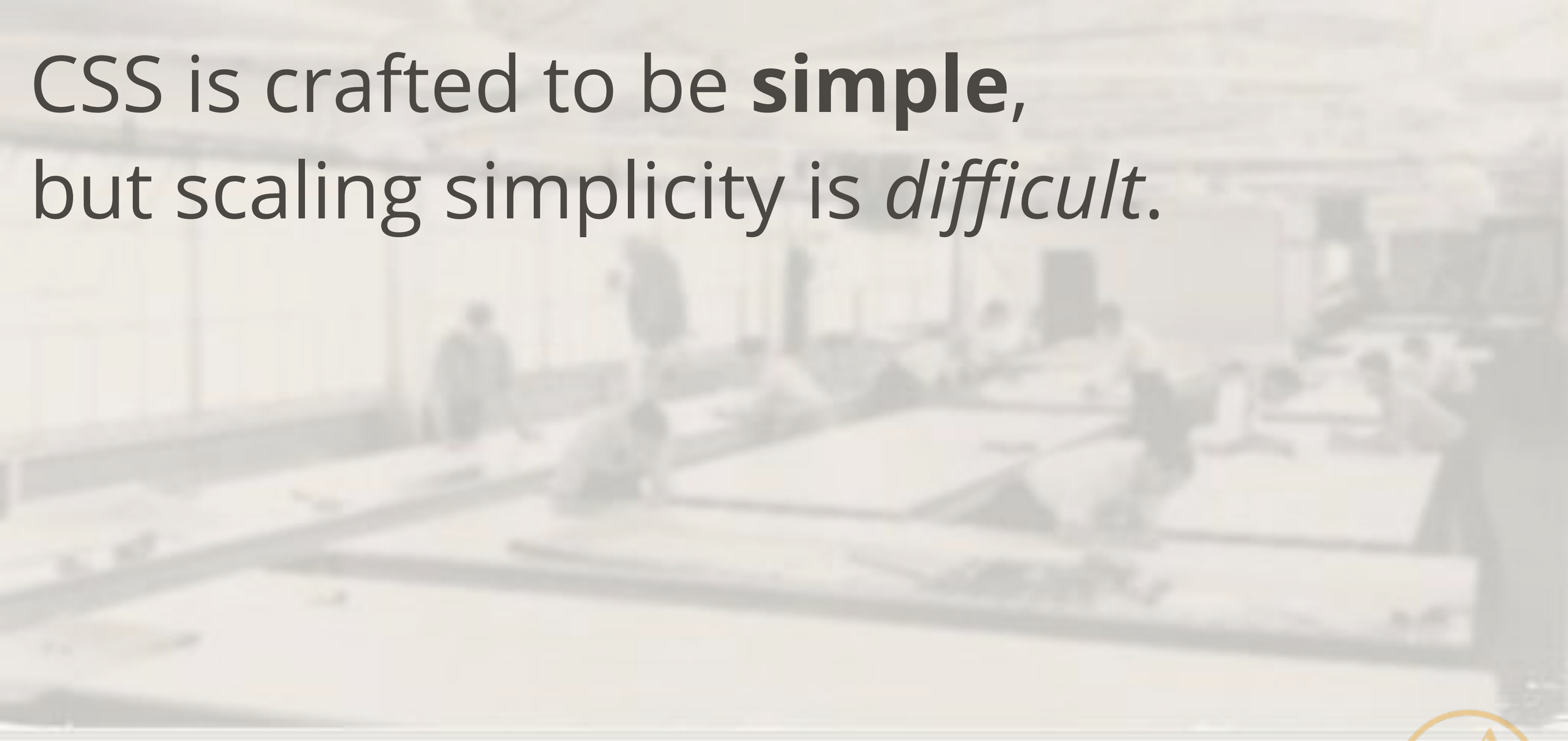
1 FOUNDATION



CSS is crafted to be **simple**,
but scaling simplicity is *difficult*.

At Scale

- Slight variations of colors, fonts, numbers, & other properties arise
- Effective curbing of repetition can decline
- Stylesheet size may become unmanageable

Enter Sass

- Syntactically Awesome Stylesheets
- Looks like CSS, but adds features to combat shortcomings
- Preprocessor, like CoffeeScript & Haml:

- Created by Hampton Catlin
- **Primary developers:**
  Nathan Weizenbaum & Chris Eppstein
- Baked into Rails

1.1 Sass, Not SASS

Assembly Tip

~~SASS~~ Sass

- Sassy CSS (`.scss`) is the default file extension
- CSS **is valid** SCSS
- A second syntax (`.sass`) exists, but we'll focus on SCSS for the course

application.scss

```scss
$main: #444;

.btn {
  color: $main;
  display: block;
}
.btn-a {
  color: lighten($main, 30%);
  &:hover {
    color: lighten($main, 40%);
  }
}
```

application.css

```css
.btn {
  color: #444444;
  display: block;
}
.btn-a {
  color: #919191;
}
.btn-a:hover {
  color: #aaaaaa;
}
```

1.2 SCSS: Sassy CSS

Assembly Tip

Since CSS doubles as valid SCSS, try writing styles normally & slowly incorporate new techniques.

- Sass adds `//` for single line comments - **not** output after compile

1.3 Commenting

application.scss

```
// These comments will
// not be output to the
// compiled CSS file

/* This comment will */
```

application.css

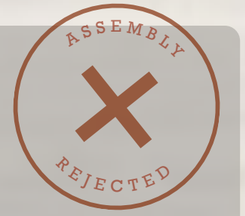
```
/* This comment will */
```

1.3 Commenting

application.css

```
/*
 *  Imports styles found in 'buttons.css'
 *  when the browser requests application.css
 */

@import "buttons.css";
```



1.4 Importing

- The CSS `@import` rule has been avoided: prevents parallel downloading
- `@import` with `.scss` or `.sass` happens **during compile** rather than client-side
- File extension is optional

1.4 Importing

application.scss

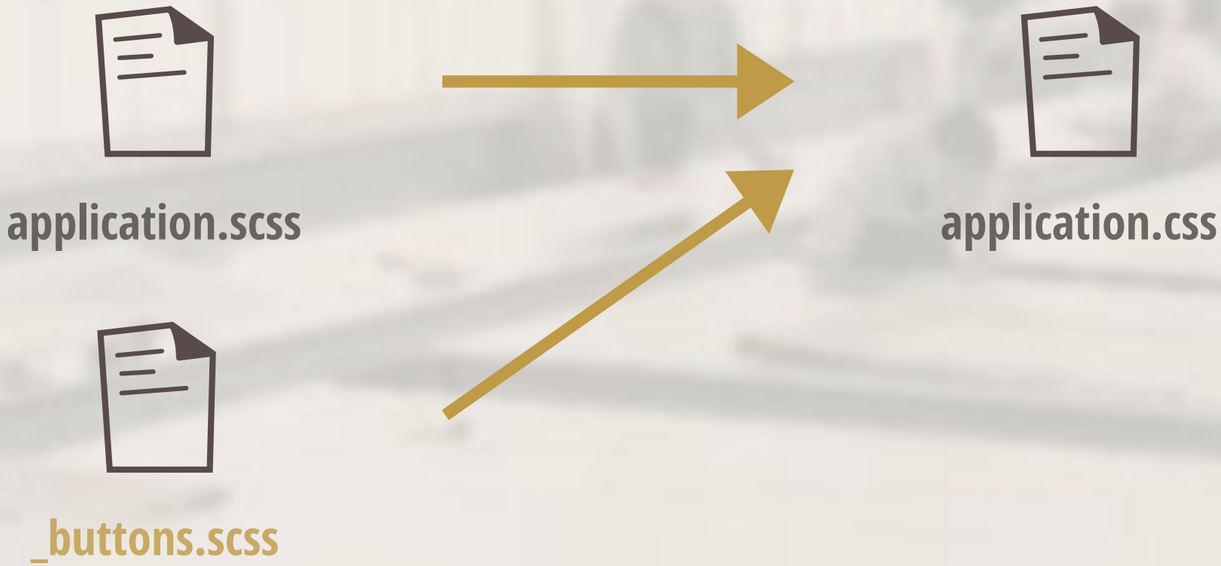
```
// Imports styles found in 'buttons.scss'
// when the compiler processes application.scss

@import "buttons";
```

1.4 Importing

Partials

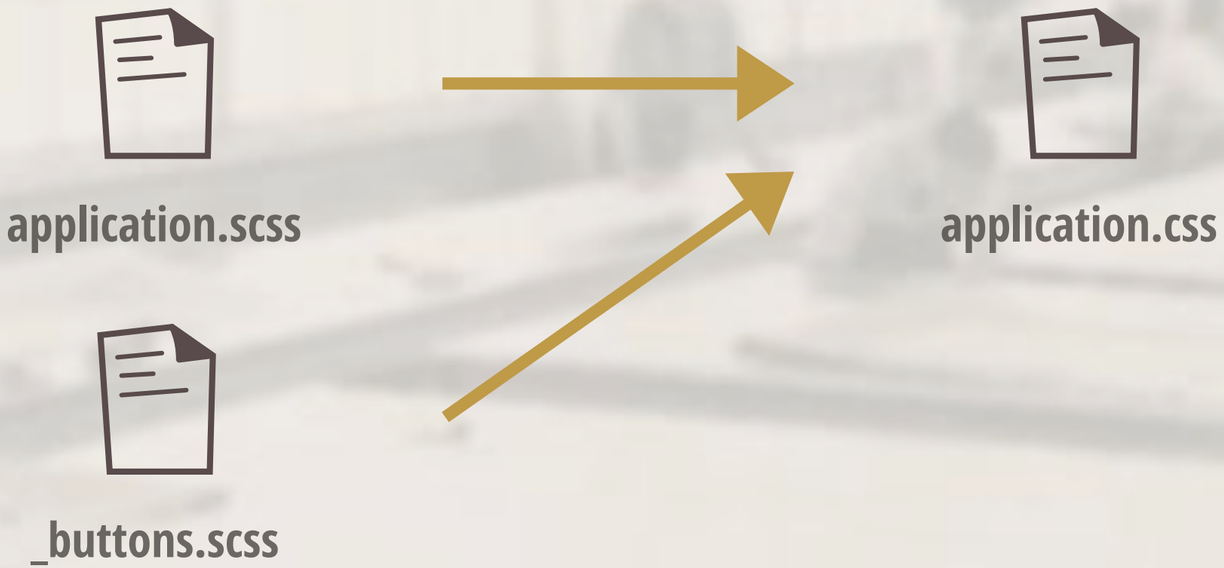
Adding an underscore creates a **partial**. Partials can be imported, but will not compile to `.css`



application.scss

```
// Will import _buttons.sass, buttons.sass,
// _buttons.scss, or buttons.scss

@import "buttons";
```



1.4 Importing

application.css


```
.content {
  border: 1px solid #ccc;
  padding: 20px;
}
.content h2 {
  font-size: 3em;
  margin: 20px 0;
}
.content p {
  font-size: 1.5em;
  margin: 15px 0;
}
```

1.5 Nesting Selectors

application.scss

```scss
.content {
  border: 1px solid #ccc;
  padding: 20px;
}
.content h2 {
  font-size: 3em;
  margin: 20px 0;
}
.content p {
  font-size: 1.5em;
  margin: 15px 0;
}
```

application.css

```css
.content {
  border: 1px solid #ccc;
  padding: 20px;
}
.content h2 {
  font-size: 3em;
  margin: 20px 0;
}
.content p {
  font-size: 1.5em;
  margin: 15px 0;
}
```

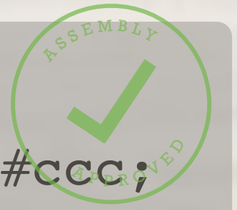
1.5 Nesting Selectors

application.scss

```scss
.content {
  border: 1px solid #ccc;
  padding: 20px;
  h2 {
    font-size: 3em;
    margin: 20px 0;
  }
  p {
    font-size: 1.5em;
    margin: 15px 0;
  }
}
```

application.css

```css
.content {
  border: 1px solid #ccc;
  padding: 20px;
}
.content h2 {
  font-size: 3em;
  margin: 20px 0;
}
.content p {
  font-size: 1.5em;
  margin: 15px 0;
}
```



1.5 Nesting Selectors

Nesting Properties

Certain properties with matching namespaces are nestable:

application.scss

```
.btn {
  text: {
    decoration: underline;
    transform: lowercase;
  }
}
```

application.css

```
.btn {
  text-decoration: underline;
  text-transform: lowercase;
}
```


While nesting, the & symbol references the parent selector:

application.scss

```
.content {
  border: 1px solid #ccc;
  padding: 20px;
  .callout {
    border-color: red;
  }
  &.callout {
    border-color: green;
  }
}
```

references:

.content

application.css

```
.content {
  border: 1px solid #ccc;
  padding: 20px;
}
.content .callout {
  border-color: red;
}
.content.callout {
  border-color: green;
}
```

application.scss

```
a {
  color: #999;
  &:hover {
    color: #777;
  }
  &:active {
    color: #888;
  }
}
```

application.css

```
a {
  color: #999;
}
a:hover {
  color: #777;
}
a:active {
  color: #888;
}
```

1.6 The Parent Selector

Parent Selector Nesting

Selectors can also be added **before** the `&` reference:

application.css

```
.sidebar {
  float: right;
  width: 300px;
}
.users .sidebar {
  width: 400px;
}
```

modifies .sidebar in a separate declaration

application.scss

```
.sidebar {
  float: right;
  width: 300px;
  .users & {
    width: 400px;
  }
}
```

references:
.sidebar

application.css


```
.sidebar {
  float: right;
  width: 300px;
}
.users .sidebar {
  width: 400px;
}
```

1.6 The Parent Selector

application.scss

```
.sidebar {
  float: right;
  width: 300px;
  h2 {
    color: #777;
    .users & {
      color: #444;
    }
  }
}
```

references:
.sidebar h2

application.css

```
.sidebar {
  float: right;
  width: 300px;
}
.sidebar h2 {
  color: #777;
}
.users .sidebar h2 {
  color: #444;
}
```

1.6 The Parent Selector

- Nesting is easy, but **dangerous**
- **Do not nest unnecessarily**

1.7 Nesting Pitfalls

application.scss

```scss
.content {
  background: #ccc;
  .cell {
    h2 {
      a {
        &:hover {
          color: red;
        }
      }
    }
  }
}
```

application.css

```css
.content {
  background: #ccc;
}
.content .cell h2 a:hover {
  color: red;
}
```

dangerous level of specificity

1.7 Nesting Pitfalls

Assembly Tip

Try limiting your nesting to 3 or 4 levels and consider refactoring anything deeper.


ASSEMBLING SASS
PROPERTY OF:
CODE SCHOOL
PROJECT DATE:
SEP 2012
STANDARD ISSUE
NO 18
CODE SCHOOL
ASSEMBLY